BDM Assignment 2

By-Shubhkumar Bharatkumar Patel

MSc-Big Data and analysis

Student ID – 3077432

Dataset: -

cosit	year	month	day	hour	minute	second	millisecond	minuteofday	lane	lanename	straddlelane	straddlelanename	class	classname	length	headway	gap	speed	weight	temperature	duration	validitycode	numberofaxles	axleweights	axlespacings
998	2021	1	29	2	15	1	0	135	1	Ch 1	0	null	2	CAR	5.1	1.57	1.44	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	1	0	135	2	Ch 2	0	null	5	HGV_RIG	11.6	1.62	1.93	71.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	2	0	135	2	Ch 2	0	null	2	CAR	5.2	1.28	1.01	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	2	0	135	1	Ch 1	0	null	5	HGV_RIG	11.1	1.17	1.53	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	4	0	135	2	Ch 2	0	null	2	CAR	5.2	1.04	1.03	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	4	0	135	1	Ch 1	0	null	2	CAR	5.1	1.04	0.72	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	5	0	135	1	Ch 1	0	null	2	CAR	5.3	1.0	1.03	70.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	5	0	135	2	Ch 2	0	null	2	CAR	5.3	1.32	1.33	71.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	7	0	135	2	Ch 2	0	null	5	HGV_RIG	11.3	1.59	1.93	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	9	0	135	1	Ch 1	0	null	2	CAR	5.1	3.26	3.23	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	9	0	135	2	Ch 2	0	null	2	CAR	5.2	1.14	0.81	71.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	10	0	135	1	Ch 1	0	null	2	CAR	5.0	1.28	1.33	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	11	0	135	2	Ch 2	0	null	5	HGV_RIG	11.3	1.3	1.64	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	12	0	135	1	Ch 1	0	null	2	CAR	5.1	1.36	1.34	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	13	0	135	2	Ch 2	0	null	5	HGV_RIG	11.5	1.24	1.31	71.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	14	0	135	1	Ch 1	0	null	5	HGV_RIG	11.3	1.31	1.73	70.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	15	0	135	1	Ch 1	0	null	2	CAR	5.0	1.54	1.22	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	15	0	135	2	Ch 2	0	null	5	HGV_RIG	11.2	1.59	1.52	69.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	17	0	135	2	Ch 2	0	null	5	HGV_RIG	11.4	1.65	1.72	71.0	0.0	0.0	0	0	0	null	null
998	2021	1	29	2	15	17	0	135	1	Ch 1	0	null	2	CAR	5.0	1.54	1.54	69.0	0.0	0.0	0	0	0	null	null

"only showing top 20 rows\n",

Q1. Calculate the usage of Irish road network in terms of percentage grouped by vehicle category.

In [7]:

```
q1=spark.sql("SELECT classname, COUNT(classname) AS count,\
round(count(classname)*100 / (select count(*) from tables),1) \
AS percentage from tables  GROUP BY classname ORDER BY percentage desc")
q1.show()
q1.write.format("org.apache.spark.sql.cassandra").\
options(table="q1", keyspace="assignment2").save(mode="append")
```

classname	count	percentage
CAR	1824109	71.2
LGV	404379	15.8
HGV_ART	182570	7.1
HGV_RIG	105864	4.1
BUS	19511	0.8
CARAVAN	16979	0.7
MBIKE	7957	0.3
null	0	0.0

```
bdm@s1: ~
bdm@s1:~$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.0.0 | Cassandra 4.0.3 | CQL spec 3.4.5 | Native protocol v5]
Use HELP for help.
cqlsh>  use assignment2;
cqlsh:assignment2> select * from q1;

 classname | count   | percentage
-----------+---------+------------
       BUS |   19511 |        0.8
   CARAVAN |   16979 |        0.7
       CAR | 1824109 |       71.2
       LGV |  404379 |       15.8
   HGV_RIG |  105864 |        4.1
     MBIKE |    7957 |        0.3
   HGV_ART |  182570 |        7.1

(7 rows)
cqlsh:assignment2>
```

Q2. Calculate the highest and lowest hourly flows on M50 - show the hours and total number of vehicle counts.

In [8]:
```
q2 = spark.sql("SELECT hour, count(hour) AS count from tables GROUP BY hour ORDER BY count desc ")
q2.show(24)
q2.write.format("org.apache.spark.sql.cassandra").options(table="q2", keyspace="assignment2").save(mode="append")
```

```
+----+------+
|hour| count|
+----+------+
|  16|226293|
|  15|218875|
|  17|211984|
|  14|196156|
|   8|195373|
|   7|190449|
|  13|182303|
|  11|151449|
|  10|141421|
|  18|132065|
|  12|131667|
|   6|113877|
|   9|104512|
|  19| 93599|
|  20| 72506|
|  21| 48493|
|   5| 39082|
|  22| 27680|
|  23| 22399|
|   4| 15991|
|   0| 15624|
|   1| 11214|
|   2|  9731|
|   3|  8797|
+----+------+
```

```
cqlsh:assignment2> select * from q2;

 hour | count
------+--------
   23 |  22399
    5 |  39082
   10 | 141421
   16 | 226293
   13 | 182303
   11 | 151449
    1 |  11214
   19 |  93599
    8 | 195373
    0 |  15624
    2 |   9731
    4 |  15991
   18 | 132065
   15 | 218875
   22 |  27680
   20 |  72506
    7 | 190449
    6 | 113877
    9 | 104512
   14 | 196156
   21 |  48493
   17 | 211984
   12 | 131667
    3 |   8797

(24 rows)
cqlsh:assignment2>
```

Q3. Calculate the evening and morning rush hours on M50 - show the hours and the total counts.

In [9]:
```
mor_rush = [8,9,10,11]
eve_rush = [17,18,19,20]
q3 = spark.sql("SELECT hour, count(hour) AS count from tables WHERE hour IN (8,9,10,11)\
GROUP BY hour ")
q3.show()
q3.write.format("org.apache.spark.sql.cassandra").\
options(table="q3", keyspace="assignment2").save(mode="append")
```

```
+----+------+
|hour| count|
+----+------+
|   9|104512|
|   8|195373|
|  10|141421|
|  11|151449|
+----+------+
```

In [10]:
```
q3eve = spark.sql("SELECT hour, count(hour) AS count from tables WHERE hour IN (17,18,19,20)\
GROUP BY hour")
q3eve.show()
q3eve.write.format("org.apache.spark.sql.cassandra").\
options(table="q3eve", keyspace="assignment2").save(mode="append")
```

```
+----+------+
|hour| count|
+----+------+
|  20| 72506|
|  19| 93599|
|  17|211984|
|  18|132065|
+----+------+
```

```
cqlsh:assignment2> select * from q3;

 hour | count
------+--------
   10 | 141421
   11 | 151449
    8 | 195373
    9 | 104512

(4 rows)
cqlsh:assignment2> select * from q3eve;

 hour | count
------+--------
   19 |  93599
   18 | 132065
   20 |  72506
   17 | 211984

(4 rows)
cqlsh:assignment2>
```

Q4.Calculate average speed between each junction on M50 (e.g., junction 1 - junction2, junction 2 - junction 3, etc.).

In [13]:
```
q4avg = spark.sql("SELECT cosit, round(AVG(speed),2) AS avgspeed from tables GROUP BY cosit")
q4avg.show()
q4junlst =[("Junction3- junction4", 1500),("Junction4-junction5", 1501),("Junction5-junction6", 1502),("Junction6-junction7",150
juncs= sc.parallelize(q4junlst).collect()
q4jun=spark.createDataFrame(juncs, ["junction","cosit"])
q4jun.show()
```

```
+------+--------+
| cosit|avgspeed|
+------+--------+
|200718|  109.76|
|  1591|   79.29|
|  1025|   92.51|
|  1507|   102.4|
|  1522|   92.62|
|  1721|   74.08|
| 31031|  110.23|
|  1303|   71.91|
|200714|   46.66|
|200722|    95.6|
|  1223|   78.81|
| 20671|   71.89|
| 20221|   93.14|
|  1016|  115.38|
| 20223|   85.68|
|  1133|   85.42|
| 20021|    93.9|
|  1331|   97.29|
|  1561|   91.42|
|200713|  102.91|
+------+--------+
only showing top 20 rows
```

```
+--------------------+-----+
|            junction|cosit|
+--------------------+-----+
|Junction3- junction4| 1500|
| Junction4-junction5| 1501|
| Junction5-junction6| 1502|
| Junction6-junction7| 1508|
| Junction7-junction9| 1503|
|Junction9-junction10| 1509|
|Junction10-juncti...| 1504|
|Junction11-juncti...| 1505|
|Junction12-juncti...| 1506|
|Junction13-juncti...| 1507|
|Junction14-juncti...|15010|
|Junction15-juncti...|15011|
|Junction16-juncti...|15012|
+--------------------+-----+
```

In [14]:
```
q4jun.registerTempTable("q4jun")
q4= spark.sql("SELECT tables.cosit,round(AVG(tables.speed),1) AS avgspeed,\
q4jun.junction from tables JOIN q4jun ON tables.cosit = q4jun.cosit \
GROUP BY tables.cosit,q4jun.junction ")
q4.show()
q4.write.format("org.apache.spark.sql.cassandra").\
options(table="q4", keyspace="assignment2").save(mode="append")
```

```
/usr/local/spark/python/pyspark/sql/dataframe.py:140: FutureWarning: Deprecated in 2.0, use createOrReplaceTempView instead.
  FutureWarning
```

```
+-----+--------+--------------------+
|cosit|avgspeed|            junction|
+-----+--------+--------------------+
|15011|   102.6|Junction15-juncti...|
| 1502|    98.0| Junction5-junction6|
| 1504|    99.7|Junction10-juncti...|
| 1508|    94.9| Junction6-junction7|
| 1505|    98.6|Junction11-juncti...|
| 1506|   101.6|Junction12-juncti...|
|15010|   105.1|Junction14-juncti...|
| 1509|    93.1|Junction9-junction10|
| 1501|    97.3| Junction4-junction5|
|15012|   105.3|Junction16-juncti...|
| 1500|    88.9|Junction3- junction4|
| 1503|    96.3| Junction7-junction9|
| 1507|   102.4|Junction13-juncti...|
+-----+--------+--------------------+
```

```
cqlsh:assignment2> select * from q4;

 cosit | avgspeed | junction
-------+----------+----------------------
  1505 |       98 | Junction11-junction12
  1500 |       88 |  Junction3- junction4
 15011 |      102 | Junction15-junction16
  1504 |       99 | Junction10-junction11
  1506 |      101 | Junction12-junction13
 15012 |      105 | Junction16-junction17
  1503 |       96 |   Junction7-junction9
  1501 |       97 |   Junction4-junction5
 15010 |      105 | Junction14-junction15
  1509 |       93 |  Junction9-junction10
  1502 |       98 |   Junction5-junction6
  1507 |      102 | Junction13-junction14
  1508 |       94 |   Junction6-junction7

(13 rows)
cqlsh:assignment2>
```

Q5. Calculate the top 10 locations with highest number of counts of HGVs (class). Map the COSITs with their names given on the map.

In [16]:
```
q5list =[("TMU M50 001.7 N", 1500),("TMU M50 005.0 N", 1501),("TMU M50 010.0 N", 1502),("TMU M50 015.0 S",1508),("TMU M50 020.0 N
q5mp= sc.parallelize(q5list).collect()
q5loclist=spark.createDataFrame(q5mp, ["location","cosit"])
q5loclist.show()
```

```
+---------------+-----+
|       location|cosit|
+---------------+-----+
|TMU M50 001.7 N| 1500|
|TMU M50 005.0 N| 1501|
|TMU M50 010.0 N| 1502|
|TMU M50 015.0 S| 1508|
|TMU M50 020.0 N| 1503|
|TMU M50 015.0 N| 1509|
|TMU M50 025.0 S| 1504|
|TMU M50 025.0 N| 1505|
|TMU M50 030.0 S| 1506|
|TMU M50 035.0 S| 1507|
|TMU M50 040.0 S|15010|
|TMU M50 035.0 N|15011|
|TMU M50 040.0 N|15012|
+---------------+-----+
```

In [17]:
```
q5loclist.registerTempTable("q5loclist")
q5 = spark.sql("SELECT  Distinct tables.cosit,  count(tables.cosit) AS count,\
q5loclist.location from tables JOIN q5loclist ON tables.cosit = q5loclist.cosit \
WHERE tables.classname ='HGV_RIG' OR tables.classname = 'HGV_ART'  \
GROUP BY tables.cosit, q5loclist.location ORDER BY count desc");
q5.show(10)
```

```
+-----+-----+---------------+
|cosit|count|       location|
+-----+-----+---------------+
| 1508| 6840|TMU M50 015.0 S|
| 1502| 6778|TMU M50 010.0 N|
| 1503| 6556|TMU M50 020.0 N|
| 1501| 6160|TMU M50 005.0 N|
| 1500| 4596|TMU M50 001.7 N|
| 1509| 2788|TMU M50 015.0 N|
| 1504| 2146|TMU M50 025.0 S|
| 1506| 1922|TMU M50 030.0 S|
| 1505| 1856|TMU M50 025.0 N|
| 1507| 1433|TMU M50 035.0 S|
+-----+-----+---------------+
only showing top 10 rows
```

In [18]:
```
q5.write.format("org.apache.spark.sql.cassandra").\
options(table="q5", keyspace="assignment2").save(mode="append")
```

```
cqlsh:assignment2> select * from q5;

 cosit | count | location
-------+-------+-----------------
  1501 |  6160 | TMU M50 005.0 N
 15010 |  1333 | TMU M50 040.0 S
  1504 |  2146 | TMU M50 025.0 S
  1502 |  6778 | TMU M50 010.0 N
  1500 |  4596 | TMU M50 001.7 N
 15011 |  1230 | TMU M50 035.0 N
  1505 |  1856 | TMU M50 025.0 N
  1503 |  6556 | TMU M50 020.0 N
  1507 |  1433 | TMU M50 035.0 S
  1509 |  2788 | TMU M50 015.0 N
  1508 |  6840 | TMU M50 015.0 S
 15012 |  1056 | TMU M50 040.0 N
  1506 |  1922 | TMU M50 030.0 S

(13 rows)
cqlsh:assignment2>
```